Real-Time GPS Spoofing Detection via Correlation of Encrypted Signals

Brady W. O'Hanlon
School of Electrical and Computer Engineering, Cornell University, Ithaca, N.Y. 14853-7501
Mark L. Psiaki
Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, N.Y. 14853-7501
Jahshan A. Bhatti, Daniel P. Shepard, and Todd E. Humphreys
Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, Texas 78712-0235

Abstract

A method for detecting the spoofing of civilian GPS signals has been implemented and successfully tested in a real-time system. GPS signal spoofing is an attack method whereby a third party transmits a signal that appears authentic but induces the receiver under attack to compute an erroneous navigation solution, time, or both. The detection system described herein provides a defense against such attacks. It makes use of correlations between the unknown encrypted GPS L1 P(Y) code signals from two narrow-band civilian receivers to verify the presence or absence of spoofing. One of these receivers is assumed to be at a secure location that is not subject to spoofing. The other receiver is the potential spoofing victim for which the present developments constitute a defense. Successful detection results are presented using a reference receiver in Ithaca, New York, a victim receiver in Austin, Texas, and a spoofer in Austin, Texas.

INTRODUCTION

GPS spoofing is a method of attacking a Global Positioning System receiver with the goal of having the targeted receiver compute an erroneous navigation solution, an incorrect time, or both [1, 2, 3]. Spoofing of the unencrypted civilian L1 C/A signal has been demonstrated both in the laboratory [2, 3] and in the field [4]. Although there have, as yet, been no confirmed GPS spoofing attacks observed "in the wild," the vulnerability of GPS to spoofing attacks has long been a concern [1, 5, 6], and an Iranian engineer claimed to have used GPS spoofing to capture a highly classified U.S. drone in December, 2011 [7]. Proposed methods for detecting spoofing attacks include examining changes to certain signal characteristics [8], incorporation of external hardware such as an inertial measurement unit [9], use of multiple [10] or moving receiver antennas [11], or cryptographic techniques [12, 13, 14, 15, 16, 17].

This work is an extension of the work presented in Ref. [13] and presents an implementation of the cryptographic technique of Refs. [12, 13, 14, 15] that operates in real-time. A major contribution of this work is that it is the first real-time implementation of a cryptographic civilian GPS spoofing detection technique. Tests of this system against a sophisticated spoofer constitute another significant contribution.

This spoofing detection technique makes use of the encrypted P(Y) code and assumes that it cannot be spoofed. Given that the U.S. military has implemented and guards this encryption, this is a reasonable assumption, with two caveats: this technique does not detect "meaconing" [17] attacks, or attacks that attempt to estimate and replay the P(Y) encryption code. The former attack is fairly limited in scope, and there are other defenses [17] against the latter attack, thus this defense still has significant value.

Given the assumed security of the P(Y) code, an attacker spoofing the civilian C/A code will be unable to provide a spoofed P(Y) code with the correct code phase and carrier phase relationship to its spoofed C/A code. The two receivers in this spoofing detection system know the expected relationship between the C/A code and the P(Y) code. They both track the C/A code and isolate the component of the signal that should be a noisy version of the P(Y) code. A communication link is used to send one of these signal components to the other receiver, which then correlates the two versions to produce a spoofing detection statistic. In this implementation, the raw unprocessed samples from one receiver front-end are transmitted to the other receiver (which then does all of the aforementioned processing), as this is more efficient when more than one satellite is being tracked.

A large value of the spoofing detection statistic indicates identical true P(Y) code in both receivers, which indicates that the potential victim receiver (the "defended receiver") is not being spoofed. A near-zero value of this statistic, on the other hand, indicates the absence of P(Y) code in one or both receivers, presumably just in the victim receiver. This low value indicates a spoofing attack. Of course, P(Y) code could be missing from the other receiver (the "reference receiver"), but this system assumes the reference receiver has been made impervious to spoofing. The choice of a detection threshold against which this statistic is compared is addressed with a hypothesis testing analysis. The hypothesis testing analysis used here is an adaptation of the one developed in Ref. [15]. Adaptation has been necessitated by simplifications in the test statistic calculation which facilitate real-time operation. The simplified statistic calculation and hypothesis test analysis constitute another contribution of this paper.

In the course of testing this algorithm, it has been necessary to perform an experimental calibration of the difference in transmitted power between the L1 C/A and P(Y) code signals. A nominal value is suggested in the GPS Interface Specification [18], but experimental results have shown that it differs slightly from the published value, and the value differs somewhat between satellites. These variations have been previously observed [19, 20], but the current satellite constellation differs significantly from its state at the time of that work. Another contribution of this paper is a table with the experimentally-derived transmission power differences for each satellite.

The remainder of the paper is organized as follows. The first section discusses the experimental set-up and the hardware used for receiving and processing the data. The second section reviews the signal model and necessary pre-processing and introduces two new loss factors that are required for correct estimation of the P(Y) signal power. Next, the algorithm for computing the spoofing test statistic and other quantities required for the hypothesis test is described. Results from testing this system under a spoofing attack are then presented. Also included in the results section is a table with the measured per-satellite difference in L1 P(Y) code vs. C/A code signal power. The paper concludes with a discussion and summary of its important points.

ARCHITECTURE

The spoofing detection system consists of two narrow-band radio front-ends and data capture systems, a full GPS software radio system with additional spoofing detection software running on a personal computer, and a method for transporting the data from the two front-ends to the software radio computer. Although the radio front-end/data capture systems are not stand-alone GPS receivers, they will nonetheless be referred to as "receivers" in the interest of brevity.

Both receivers perform mixing, filtering, and sampling of the radio-frequency signal, producing 2-bit samples at a sample rate of 5.7 MHz. The intermediate frequency filters in the two front-ends each have a nominal 3-dB bandwidth of 2.5 MHz. This narrow bandwidth attenuates the wide-band P(Y) signal by 5.5 dB and distorts it. The exact filter response for each receiver has been characterised using off-line system identification techniques [21]. Knowledge of this response is important for the correct design of a spoofing detection hypothesis test that properly accounts for the P(Y) signal attenuation and distortion.

The data from each receiver are either directly recorded to a hard drive, or streamed over a network connection and then recorded to a hard drive, depending on the particular equipment set-up. The software receiver and spoofing detection calculations operate on the data that is stored to disk. It can operate in a real-time mode or an after-the-fact mode. It would be possible to replace the disk storage with a buffer in RAM.

To reduce the possibility of a man-in-the-middle type attack where the data transmitted over the network are intercepted by a third party and replaced with other (possibly spoofed) data, a secure shell (SSH) tunnel was used for all network communication. This encrypts the data using triple DES encryption and is believed to be secure [22].

Software that carries out the spoofing detection calculations has been added to an existing GPS software receiver [23]. This software processes the samples from both front-ends in parallel. It tracks the civilian L1 C/A code signals, performs P(Y) code cross-correlation, and produces a metric that indicates the likelihood that the defended receiver is being spoofed. This software receiver could be physically located anywhere; for this system it was co-located with the reference receiver for convenience and the raw unprocessed samples from the defended receiver were transmitted to it over the Internet. By transmitting the unprocessed samples rather than the processed signal component for each individual signal, the required communications bandwidth is reduced. The receiver architecture used for this work is illustrated in Fig. 1. Except for the SSH tunnel and the fact that the software receiver simultaneously processes data from two radio front-ends, everything to the left of the dashed vertical line represents standard software radio hardware and code. Everything to the right of that line represents new calculations needed for spoofing detection.

All processing for this work was done on a personal computer with a quad-core Intel i7 930 CPU. This implementation processes data at approximately 3 times faster than real-time (i.e., it processes 30 seconds of data in 10 seconds); up to 30 satellites common to both receivers can be tracked simultaneously in real-time. Standard hemispherical patch antennas were used at both the reference and defended receivers.

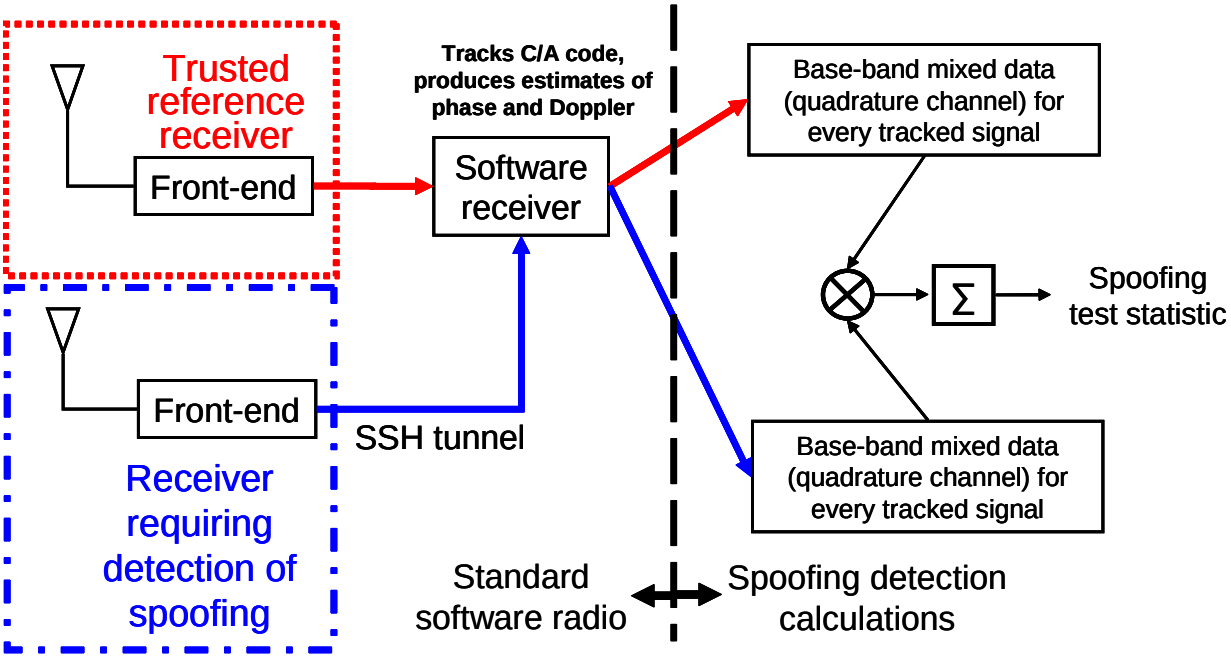

Fig. 1: An example receiver architecture.

Signal Model and Pre-Processing

The pre-processing and signal model used here are developed in Ref. [15]. For convenience, the relevant subset of equations from Ref. [15] are reproduced in this section.

Signal Model

The models of the signals at the outputs of the RF fronts ends of the two receivers take the form:

$$y_{Ai} = A_{cA}C_f(t_{Ai})D(t_{Ai})\cos[\omega_{IF}t_{Ai} + \phi_A(t_{Ai})] + A_{pA}P_{Yf}(t_{Ai})D(t_{Ai})\sin[\omega_{IF}t_{Ai} + \phi_a(t_{Ai})] + n_{Ai} \quad (1a)$$

$$y_{Bi} = A_{cB}C_f(t_{Bi})D(t_{Bi})\cos[\omega_{IF}t_{Bi} + \phi_B(t_{Bi})] + A_{pB}P_{Yf}(t_{Bi})D(t_{Bi})\sin[\omega_{IF}t_{Bi} + \phi_B(t_{Bi})] + n_{Bi} \quad (1b)$$

where y_{Ai} is the sample output by the RF front-end of receiver A, the reference receiver, at receiver A clock time t_{Ai}, and y_{Bi} is the output of receiver B, the defended receiver, at receiver B clock time t_{Bi}.

$C_f(t)$ and $P_{Yf}(t)$ are functions representing the C/A and P(Y) codes, respectively, after attenuation and distortion by the RF front-end. The function $D(t)$ represents the 50 Hz navigation data bit modulation. A_{cA} and A_{cB} are the received C/A code amplitudes for receivers A and B, and the corresponding P(Y) code amplitudes are A_{pA} and A_{pB}.

The nominal intermediate frequency is ω_{IF}, and $\phi_A(t)$ and $\phi_B(t)$ are the beat carrier phase time histories of the signals at the two receivers. The time derivatives of $\phi_A(t)$ and $\phi_B(t)$ equal the receiver carrier Doppler shift.

The remaining terms, n_{Ai} and n_{Bi}, are the receiver noise terms, which are assumed to be discrete-time Gaussian

white-noise with statistics:

$$E(n_{Ai}) = 0, E(n_{Ai}^2) = \sigma_{RFA}^2, E(n_{Ai}n_{Aj}) = 0 \text{ for all } i \neq j \tag{2a}$$

$$E(n_{Bi}) = 0, E(n_{Bi}^2) = \sigma_{RFB}^2, E(n_{Bi}n_{Bj}) = 0 \text{ for all } i \neq j \tag{2b}$$

$$E(n_{Ai}n_{Bi}) = 0 \text{ for all } i, j \tag{2c}$$

Tracking the C/A Signal

To process the C/A code, the signal from the front end is mixed with carrier and code replicas and accumulated for some period. These accumulations are then used in discriminators in a Delay-Lock-Loop (DLL) and a Phase-Lock-Loop (PLL) for feedback-based tracking. The prompt in-phase and quadrature accumulations for the k^{th} accumulation interval are:

$$I_k = \sum_{i=i_k}^{i_k+N-1} y_i C[(i_k\delta T - \tau_k)(1 + \hat{\omega}_{Dk}/\omega_{L1})] \times \cos(\omega_{IF} i_k \delta T + \hat{\phi}_k + \hat{\omega}_{Dk}(i_k\delta T - \tau_k)) \tag{3a}$$

$$Q_k = \sum_{i=i_k}^{i_k+N-1} y_i C[(i_k\delta T - \tau_k)(1 + \hat{\omega}_{Dk}/\omega_{L1})] \times \sin(\omega_{IF} i_k \delta T + \hat{\phi}_k + \hat{\omega}_{Dk}(i_k\delta T - \tau_k)) \tag{3b}$$

where , δT is the nominal front-end sampling period, τ_k is the DLL-produced estimate of the start time of the first code period in the accumulation as measured by the receiver clock, and the sample index i_k is the first sample in that C/A code period (i.e., the first sample such that $i_k\delta T \geq \tau_k$). The number of samples in the accumulation is N_k, which can be approximated as N, a constant. The function $C[t]$ is the local replica of the PRN code without RF filter effects; it takes on the values ± 1. The PLL's carrier Doppler shift estimate for the k^{th} code period is $\hat{\omega}_{Dk}$, and $\hat{\phi}_k$ is the estimated beat carrier phase at the code period start time τ_k. The subscripts A and B have been omitted here as the processing is identical for both receivers.

The spoofing detection algorithm exploits the known phase-quadrature relationship of the encrypted P(Y) code relative to the C/A code, as per Eq. 1. It isolates the quadrature part of the signal with the help of the PLL and DLL outputs $\hat{\phi}$, $\hat{\omega}_{Dk}$, and τ_k. The C/A code accumulations in Eq. 3 are used to estimate the carrier-to-noise ratios required by the hypothesis test.

Received Signal Power

The received carrier-to-noise ratio of the P(Y) signals in the two receivers are important inputs to the spoofing detection calculations. These quantities can be inferred based on measured C/A code carrier-to-noise ratios coupled with knowledge of receiver properties and calibrated relationships between the transmitted C/A code power and P(Y) code power for the various GPS satellites.

The inputs to the C/A carrier-to-noise calculation are time histories of the I_k and Q_k prompt accumulations. These values can be used to estimate the amplitude of the [I_k;Q_k] vector and the variance of the Gaussian noise

in each individual I_k and Q_k accumulation. These estimates are:

$$A_{IQ} = (\bar{z}^2 - \sigma_z^2)^{1/4} \tag{4a}$$

$$\sigma_{IQ}^2 = 0.5(\bar{z} - \sqrt{\bar{z}^2 - \sigma_z^2}) \tag{4b}$$

where $\bar{z}$ is the mean of the accumulation power and σ_z^2 is its variance. These are calculated from the raw accumulations as follows:

$$\bar{z} = \frac{1}{K}\sum_{k=1}^{K}(I_k^2 + Q_k^2) \cong E\{I_k^2 + Q_k^2\} \tag{5a}$$

$$\sigma_z^2 = \frac{1}{K}\sum_{k=1}^{K}(I_k^2 + Q_k^2)^2 - \bar{z}^2 \cong E\{[I_k^2 + Q_k^2]^2\} - \bar{z}^2 \tag{5b}$$

where K is the number of prompt accumulations included in these averages. For this work, $K = 1000$ has been used with one millisecond accumulations. These calculations can be used to estimate the effective variance of the noise in the raw RF samples:

$$\sigma_{RF}^2 = \frac{2}{N}\sigma_{IQ}^2 \tag{6}$$

This notation differs somewhat from Ref. [15] because here the number of samples in each accumulation, N, is constant, whereas in that work it was allowed to vary.

The C/A code carrier-to-noise ratio is computed using the quantities in Eq. 4 as:

$$(C/N_0)_c = \frac{A_{IQ}^2}{2\sigma_{IQ}^2 T_{accum}} \tag{7}$$

where $T_{accum} = \Delta t N$ is the accumulation period.

The P(Y) code carrier-to-noise ratio can be computed from the C/A carrier-to-noise ratio, but several loss factors must be taken into account. These loss factors account for the effect of using un-filtered C/A code in Eq. 3 and the effects of the front-end filtering on the C/A signal. Let the combined effect of these two influences be denoted L_{fca}. The effect of the RF filter on the received P(Y) signal produces the loss L_{fpy}. An additional loss factor, L_{psv}, represents the difference in transmitted power between the P(Y) and C/A signals. This is nominally 3 dB, with P(Y) power lower than C/A power [18]. It has been discovered to vary between satellites and to differ somewhat from the nominal value. A contribution of this work is the experimental calibration of this previously unknown variation between satellites.

The P(Y) code carrier-to-noise ratio is:

$$(C/N_0)_{py} = L_{psv} L_{fpy} \left[\frac{10^{-0.04/10}(C/N_0)_c}{L_{fca}}\right] \tag{8}$$

The method of calculating L_{fpy}, L_{fca}, and the power of 10 are explained in Ref. [15]. The values for L_{psv} are given in a table in the Results section. Excluding the contribution of L_{psv}, the factor $L_{fpy}\frac{10^{-0.04/10}}{L_{fca}}$ has been computed to be -5.06 dB and -4.92 dB for the reference and defended receivers, respectively.

METHODOLOGY

This section explains the real-time implementation of the spoofing detection technique presented in Refs. [12, 13, 14, 15]. Several of the modifications needed for practical real-time implementation are non-trivial.

Spoofing Detection Statistic Calculation

To detect if the defended receiver is being spoofed, a spoofing detection statistic as in Ref. [15] has been produced. To accomplish this, the part of the signal from each receiver that is in quadrature with the C/A code is isolated, and the two quadrature data streams from the two receivers are mixed together and accumulated. Doing this requires knowledge of the carrier Doppler shift $\hat{\omega}_D$, carrier phase $\hat{\phi}$, and code start time estimate τ for each signal from both receivers. It also requires the data from both receivers to have been aligned in time, i.e., a given pair of mixed quadrature samples from the two receivers must have the same time of transmission.

To accomplish time alignment, the software first performs a coarse synchronization by locating the beginning of the same C/A code period in the data from the two receivers for each signal. This is done by decoding the transmitted data bits from each receiver to calculate the GPS time of week which indicates the transmission time of a particular code period [18]. Each channel then processes however many C/A code periods worth of data are required such that at the end of this process the reference and defended receiver channels are all processing the C/A code period corresponding to the same nominal transmit time (as this is a software receiver, "channel" in this context means the code that handles the signal from a particular satellite). In general this requires a small amount of waiting for the desired data to become available, depending on how closely synchronized the data gathering is at the two receivers, on the receiver-satellite geometry, and on network latency. The delay is generally a small fraction of a second, so the process can still be considered "real-time."

After this coarse synchronization, the reference and defended receiver channels each process a single C/A code period, followed by a P(Y) code cross-correlation for the same period. To perform this cross-correlation, the quadrature part of the signal from each channel must be isolated, and a set of samples from the reference receiver must be somehow matched to a set of samples from the defended receiver. This sample matching is, in effect, a fine time alignment between the two receivers that attempts to match samples as closely as possible by the time of transmission of the underlying waveform. As the receiver is tracking the phase of the L1 C/A signal with a PLL in the normal course of operation, the correct carrier phase $\hat{\phi}$ and frequency $\hat{\omega}_D$ required for quadrature mixing are known.

The code start time estimate τ is also known for each receiver, but these times do not directly indicate the optimal matching of samples between the two receivers for cross-correlation. To determine the correct sample matching between the two receivers it is necessary to consider the effects of relative motion between the satellites and those receivers as it affects their relative Doppler shifts. In general, the carrier Doppler shifts of the signals from each receiver will differ. To produce a P(Y) code cross-correlation with maximum power, the nominal alignment of data between the two receivers is selected such that the midpoint of their C/A code periods are aligned. This causes the P(Y) codes to be aligned as well.

It should be noted that in the software receiver implementation many variables are stored as fixed-point for purposes of processing efficiency. In particular, the code start time estimate τ is stored as an integer sample index

i_k plus a fractional sample index i_{k-frac}:

$$\tau = \Delta T \left(i_k + \frac{i_{k-frac}}{S_T} \right) \tag{9}$$

where ΔT is the front-end sampling period. The variable i_{k-frac} is always non-negative (though it is stored as a signed quantity), and scaled by a factor S_T such that 1 sample has S_T subdivisions. In general, a given C/A code period has initial sample indices i_{kA} and i_{kB} for receivers A and B, respectively. It also has fractional sample counts $i_{kA-frac}$ and $i_{kB-frac}$. Also note that cross-correlation is done a single C/A code period at a time (a "sub-accumulation"), and these sub-accumulations are summed over many code periods to produce the final result.

In the simplest case, $i_{kA-frac}$ and $i_{kB-frac}$ are equal, and the carrier Doppler shifts in the two receivers are identical. Sample index matching is simple in this case: the quadrature sample with index i_{kA} in receiver A is mixed with the quadrature sample with index i_{kB} from receiver B, sample $i_{kA}+1$ in receiver A is mixed with sample $i_{kB}+1$ in receiver B, and so on. The general case of non-equal i_{k-frac} values and Doppler shifts requires more consideration.

In correcting for the signal Doppler shift, an "effective code start time" is calculated. This effective code start time indicates the sample time that is, considering the effect of Doppler shift, exactly one-half of a C/A code period away from the midpoint of the accumulation interval. This correction manifests as a correction to the fractional sample index i_{k-frac} as follows. Given the number of samples in the code-period accumulation, N, the nominal L1 frequency ω_{L1}, the signal Doppler shift frequency, $\hat{\omega}_D$, the sample period ΔT, and the nominal C/A code chipping rate F_{c-nom}, the correction based on the fractional code start time estimate is:

$$\delta i_{k-frac} = round \left(\frac{N\Delta T \left[F_{c-nom} \left(1 + \frac{\hat{\omega}_D}{\omega_{L1}} \right) - F_{c-nom} \right] S_T}{2 F_{c-nom} \Delta T} \right) = round \left[\frac{N \hat{\omega}_D S_T}{2 \omega_{L1}} \right] \tag{10}$$

For $N\Delta T \approx 1$ millisecond and a stationary receiver, $|\delta i_{k-frac}| << S_T$. This correction is subtracted from the current fractional code start time to give an effective fractional code start time:

$$\hat{i}_{k-frac} = i_{k-frac} - \delta i_{k-frac} \tag{11}$$

Note that this new $\hat{i}_{k-frac}$ could be negative. If $\hat{i}_{k-frac} < 0$, $\hat{i}_{k-frac}$ is incremented by S_T and the effective integer sample index is $\hat{i}_k = i_k - 1$. If $\hat{i}_{k-frac} > S_T$, $\hat{i}_{k-frac}$ is decremented by S_T and $\hat{i}_k = i_k + 1$. Otherwise the effective integer sample index is $\hat{i}_k = i_k$. Equations 10 and 11 are applied to the data from receivers A and B, producing $\hat{i}_{k-frac-A}$, $\hat{i}_{k-A}$, $\hat{i}_{k-frac-B}$, and $\hat{i}_{k-B}$.

Interpolation is required to yield the same or nearly the same transmission times of the mixed P(Y) codes from the two receivers. Otherwise, a large loss of correlation power will occur due to the high chipping rate of the P(Y) code (10.23 MHz). In Ref. [15] a linear interpolation of the samples from one receiver is performed such that the data from both receivers were at the same effective transmission times. This type of interpolation is avoided to minimize real-time computing requirements and a simple nearest neighbor interpolation is performed instead. This interpolation has the effect of choosing the set of samples from each receiver such that at the midpoint of the accumulation interval there is a maximum transmission time offset of $\frac{\Delta T}{2}$ between the C/A codes from the

two receivers. The effective code start times introduced in connection with Eqs. 10 and 11 are used to perform the sample matching needed for this interpolation.

To determine the first sample to use in sub-accumulation, the difference of the two effective fractional code start times is taken:

$$\Delta\hat{i}_{k-frac} = \hat{i}_{k-frac-A} - \hat{i}_{k-frac-B} \tag{12}$$

The index of the first sample for each sub-accumulation from each receiver is then:

$$\tilde{i}_{k-A} = \hat{i}_{k-A} + \max\left[\text{round}\left(\frac{\Delta\hat{i}_{k-frac}}{S_t}\right), 0\right] \tag{13a}$$

$$\tilde{i}_{k-B} = \hat{i}_{k-B} - \min\left[\text{round}\left(\frac{\Delta\hat{i}_{k-frac}}{S_t}\right), 0\right] \tag{13b}$$

This nearest-neighbor interpolation may cause a loss in correlation power due to residual P(Y) code mis-alignment between the two receivers. Another contribution of this work is calculation of an additional loss factor that accounts for any such reduction in cross-correlation power. This loss factor is denoted L_{pxc}, and its value depends on the shape of the autocorrelation function of the P(Y) code after it is filtered by the receivers' front-ends. It also depends on the magnitude of the code offset error during each sub-accumulation interval. This error for each sub-accumulation is

$$\hat{i}_{err} = \begin{cases} \frac{\Delta\hat{i}_{k-frac}}{S_T} & \text{if } |\Delta\hat{i}_{k-frac}| \leq \frac{S_T}{2} \\ 1 - \frac{\Delta\hat{i}_{k-frac}}{S_T} & \text{if } \Delta\hat{i}_{k-frac} > \frac{S_T}{2} \\ 1 + \frac{\Delta\hat{i}_{k-frac}}{S_T} & \text{if } \Delta\hat{i}_{k-frac} < \frac{-S_T}{2} \end{cases} \tag{14}$$

and is measured in units of samples.

To determine the shape of the filtered P(Y) autocorrelation function, the RF front-end filter response function has been estimated using off-line system identification techniques as in Ref. [21]. Then, using this response and a simple triangular model for the un-filtered P(Y) code autocorrelation function, the real part of the filtered P(Y) code autocorrelation function has been calculated. Figure 2 plots both the original wide-band P(Y) code autocorrelation function (green dash-dotted line) and the filtered P(Y) code autocorrelation function (solid blue line).

In performing the cross-correlation required to detect the presence of spoofing, the instantaneous cross-correlation power loss equals the value of the filtered autocorrelation function of Fig. 2 evaluated at the offset between the P(Y) codes from the two receivers. Consistent with Eq. 14, the maximum code misalignment error is $\pm\frac{1}{2}$ of a front-end sample. This equals ± 0.895 P(Y) code chips. These maximum offsets are shown in Fig. 2 as the vertical dashed red lines. Therefore, the worst case power loss factor caused by nearest-neighbor interpolation is 0.87 (-0.6 dB). This small loss justifies the decision to use nearest neighbor interpolation in order to simplify the real-time signal processing.

If the spoofing detection statistic were calculated over a single C/A code period, the correlation power loss would be calculated by evaluating the filtered autocorrelation function at the code alignment error value $\hat{i}_{err}$. Because the spoofing detection statistic is calculated by summing over many C/A code periods, an alternate approach is used. The correlation power is computed for each individual period, and the results averaged over the full accumulation

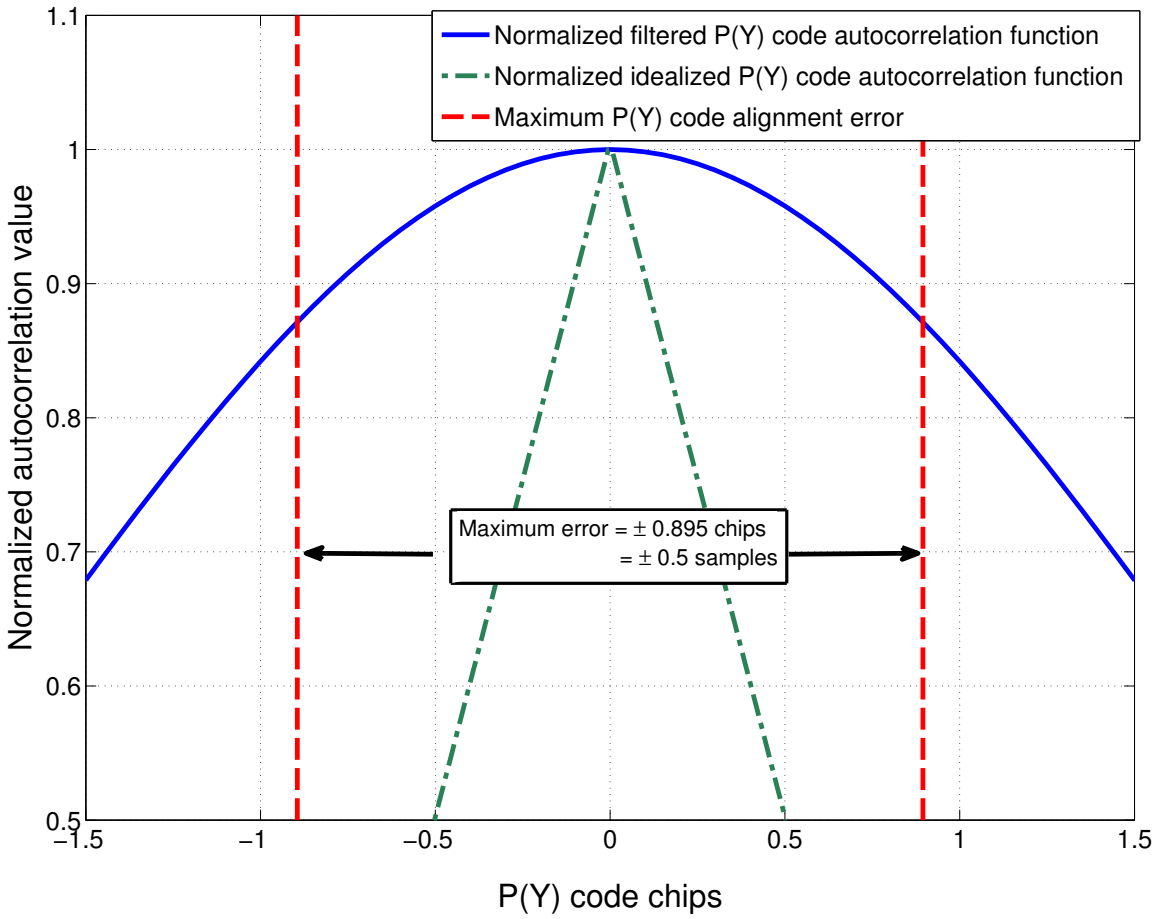

Fig. 2: Filtered P(Y) code normalized autocorrelation function.

period of the detection statistic to produce L_{pxc}. In this implementation, the filtered autocorrelation function values are stored in a look-up table. This value L_{pxc} is used as part of the spoofing detection hypothesis calculations in the following subsection.

Once the correct sample index has been selected for the start of the sub-accumulation interval, carrier mixing must be performed. Note that at this point in the processing, the receiver has already performed carrier mixing in order to compute the C/A code in-phase and quadrature accumulations, I_k and Q_k, as per Eq. 3. If this data that had been previously mixed with the quadrature carrier replica were buffered, no additional work would be necessary. The software would select the portion of this carrier-mixed data that begins at sample index $\tilde{i}_k$ from each receiver and use that in the cross-correlation. Rather than buffer the data, in this implementation the carrier replica is again mixed with the data, starting at the chosen index. This was a design decision made to allow more flexibility in the prototype system.

Given the PLL's carrier phase estimate $\hat{\phi}$, the start index of the sub-accumulation period $\tilde{i}_k$, and the carrier

Doppler shift $\hat{\omega}_D$, a quadrature carrier replica is generated using a technique based on Ref. [24], and the quadrature part of the signal is isolated. For the k^{th} code period, the quadrature base-band mixed signal is

$$y_{qi} = y_i \sin\left[\omega_{IF} t_i + \hat{\phi}_k + \hat{\omega}_{Dk}\left(t_i - \tau\right)\right] \text{for } t_i = \delta T[\tilde{i_k}, ..., (\tilde{i_k} + N - 1)] \tag{15}$$

This quantity is produced for each receiver, after which the un-normalized spoofing detection statistic is computed. This statistic is simply:

$$\gamma_u = \sum_{k=0}^{M-1} \sum_{\Delta i=0}^{N-1} y_{qA(\tilde{i}_{k-A}+\Delta i)} y_{qB(\tilde{i}_{kB}+\Delta i)} \tag{16}$$

where M is the number of C/A code periods in the accumulation, and $y_{qA(\tilde{i}_{k-A}+\Delta i)}$ and $y_{qB(\tilde{i}_{k-B}+\Delta i)}$ are the quadrature base-band data samples at sample index $\tilde{i}_{k-A}+\Delta i$ and $\tilde{i}_{k-B}+\Delta i$ from receivers A and B, respectively. This implementation differs somewhat significantly from that in Ref. [15]. In that work, the in-phase and quadrature accumulations I_k and Q_k were used to correct for any PLL tracking errors. That work made the assumption that noise effects on I_k and Q_k were negligible, which for this work proved not to be the case. Producing the detection statistic γ_u as prescribed in Ref. [15] led to a result with less power than simple multiplication by the quadrature carrier replica.

The actual implementation of Eqs. 15 and 16 is slightly different than the definitions of the quantities imply for reasons of computational efficiency. The receivers use 2-bit quantization and bitwise parallel operations [25]. This means that the data and the local carrier and code replicas are all quantized to two bits, with one bit indicating the sign of element, and the other bit indicating the magnitude. 32 consecutive elements are then stored in two 32-bit integers, with all of the sign bits stored in one integer, and all the magnitude bits stored in the other. This allows processing of 32 samples in parallel. Mixing of various elements (e.g., mixing the carrier replica with the data) involves only shifts and logical operations on the 32-bit integers. This approach has been exploited in order to design an efficient algorithm for performing cross-correlation between the two data streams.

First, each data stream is partially mixed with its respective carrier replica. This partial carrier mixing does two things: it shifts the data so that the first bit of the first 32-bit word in the accumulation interval corresponds to the desired sample index $\tilde{i}_{k-A}$ (or $\tilde{i}_{k-B}$), and it multiplies the sign bits of the carrier replica with the sign bits of the data. The results of this operation from the two receivers are then multiplied together. These multiplies are done via the exclusive-or operation. The rest of the cross-correlation is done with a look-up table, with the input being a 20-bit word composed of 4 bits of the multiplied carrier sign bits and data sign bits, 4 data magnitude bits from each receiver, and 4 carrier replica magnitude bits from each receiver. The look-up table output is the integer that results from the multiply-and-accumulate of the 4 samples from each receiver and their respective carrier replicas. This result is accumulated for the desired integration time (i.e., a total of $M * N$ samples, from Eq. 16), and produces the un-normalized spoofing detection statistic γ_u. For a good description of similar bit-wise parallel software radio calculations, see Ref. [25].

Note that the bit-wise parallel sine and cosine replicas used in Eqns. 3 and 15 have amplitudes greater than one. The effect of these non-unit amplitudes divide out of the final normalized spoofing detection statistic presented in

the next subsection.

Spoofing Detection Hypothesis Test

In addition to the un-normalized spoofing detection statistic γ_u, several other quantities must be calculated in order to implement a precise hypothesis test. Following directly from Ref. [15], two hypotheses are presented: H_0, the hypothesis that the receiver is free of spoofing, and H_1, the hypothesis that the receiver is being spoofed. The mean and variance of the spoofing detection statistic γ_u under H_1 are:

$$\bar{\gamma}_{u|H1} = 0 \tag{17}$$

$$\sigma^2_{\gamma u|H1} = \frac{MN}{4}\sigma^2_{RFA}\sigma^2_{RFB}\left[1 + 2\Delta T(C/N_0)_{pyA}\right] \tag{18}$$

The mean value under this hypothesis is zero because if the receiver is being spoofed, the effective received P(Y) code power is zero, thus the cross-correlation produces no power.

Under H_0, the spoofing detection statistic mean and variance are:

$$\bar{\gamma}_{u|H0} = MN\sigma_{RFA}\sigma_{RFB}\Delta T L_{pxc}\sqrt{(C/N_0)_{pyA}(C/N_0)_{pyB}} \tag{19}$$

$$\sigma^2_{\gamma u|H0} = \frac{MN}{4}\sigma^2_{RFA}\sigma^2_{RFB}\{1 + 2\Delta T[(C/N_0)_{pyA} + (C/N_0)_{pyB})]\} \tag{20}$$

where L_{pxc} is the cross-correlation loss factor introduced in the previous subsection.

These terms are then normalized by the standard deviation under the hypothesis of spoofing, H_1, resulting in these means and standard deviations:

$$\bar{\gamma}_{norm|H1} = 0 \tag{21}$$

$$\sigma_{\gamma norm|H1} = 1 \tag{22}$$

$$\bar{\gamma}_{norm|H0} = 2\Delta T L_{pxc}\sqrt{\frac{MN(C/N_0)_{pyA}(C/N_0)_{pyB}}{1 + 2\Delta T(C/N_0)_{pyA}}} \tag{23}$$

$$\sigma_{\gamma norm|H0} = \sqrt{\frac{1 + 2\Delta T[(C/N_0)_{pyA} + (C/N_0)_{pyB})]}{1 + 2\Delta T(C/N_0)_{pyA}}} \tag{24}$$

The same normalization must be applied to the spoofing detection statistic that has been calculated from the quadrature samples:

$$\gamma_{norm} = \frac{\gamma_u}{\sigma_{RFA}\sigma_{RFB}\sqrt{\frac{MN}{4}\left[1 + 2\Delta T(C/N_0)_{pyA}\right]}} \tag{25}$$

Assuming the probability density functions under the spoofed and un-spoofed hypotheses, $p(\gamma_{norm}|H_1)$ and $p(\gamma_{norm}|H_0)$, are Gaussian [15], and given a probability of false alarm α_{fa}, a spoofing detection threshold γ_{th} can be computed by solving the following equation:

$$\begin{aligned}\alpha_{fa} &= \int_{-\infty}^{\gamma_{th}} p(\gamma_{norm}|H_0) d\gamma_{norm} \\ &= \frac{1}{\sqrt{2\pi}\sigma_{\gamma norm|H0}} \int_{-\infty}^{\gamma_{th}} \exp\left[-\frac{(\gamma_{norm} - \bar{\gamma}_{norm|H0})^2}{2\sigma^2_{\gamma norm|H0}}\right] d\gamma_{norm}\end{aligned} \tag{26a}$$

In practice, this equation is solved off-line by assuming a zero-mean, unit-variance distribution and choosing a small value of α_{fa} ($\alpha_{fa} = 0.0001$ for all results herein), and the result is embedded in the source code. During processing, this result is then multiplied by $\sigma_{\gamma norm|H0}$ and then added to $\bar{\gamma}_{norm}$ to produce the threshold γ_{th} required for hypothesis testing. The actual output of the receiver is

$$\tilde{\gamma} = \gamma_{norm} - \gamma_{th} \tag{27}$$

for each of the satellites that are common to both receivers A and B. If $\tilde{\gamma} > 0$ for a particular satellite, it is determined that that particular signal is not being spoofed. If $\tilde{\gamma} \leq 0$, that channel is marked as being spoofed.

The probability of successfully detecting a spoofing attack is

$$\begin{aligned}P_d &= \int_{-\infty}^{\gamma_{th}} P(\gamma_n|H_1)\, \mathrm{d}\gamma_n \\ &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\gamma_{th}} \exp(-0.5\gamma^2_{norm}) d\gamma_{norm}\end{aligned} \tag{28a}$$

Note that P_d depends on the spoofing detection statistic threshold, which itself depends on the the integration time, on the noise variance, and on the signal carrier-to-noise ratio. The latter quantity varies with time due to environmental changes such as satellite elevation. Rather than aiming for a fixed P_d and adjusting the integration time to account for variations in carrier-to-noise ratio, a fixed integration time has been chosen for all signals, regardless of their carrier-to-noise ratio. For this implementation, a nominal integration time of 2 seconds has been used. For a C/A code carrier-to-noise ratio of 50 dB-Hz at both the reference and defended receivers, for a P(Y) power transmission decrement of L_{psv} of 3 dB, for a false alarm probability of 0.01%, and for a 2 second integration time, P_d is greater than 99.999%.

Although P_d varies for each signal, α_{fa} is always fixed and small, so any indication of spoofing is a reliable indicator that that signal is being spoofed.

RESULTS

Several tests have been conducted using this implementation of codeless spoofing detection. For these tests, a receiver located on the roof of a building in Ithaca, New York (42.44° N, 76.48° W) was used as the reference receiver, and a receiver located on the roof of a building in Austin, Texas (30.29° N, 97.74° W) was used as the defended receiver.

Determining The Per-Satellite Power Variation

Initial results showed some disagreement between the expected value of the spoofing test statistic and its actual value, which varied by satellite. It was theorized that, for some satellites, the difference in power between the transmitted C/A code signal and the transmitted P(Y) code signal varied from the 3 dB decrement suggested by Ref. [18]. To explore this possibility, data were collected from both receivers every half hour on February 7, 8, and 12, 2013. It was assumed that both receivers were free of spoofing during this period (and indeed, none was detected). It was also assumed that the actual transmission power decrement L_{psv} was exactly 3 dB, and a correction to this value was solved for. These data were processed using the above algorithms, with $\bar{\gamma}_u$ and γ_u calculated over two second accumulation intervals for the length of each data set, which was nominally 150 seconds. The mean values of $\bar{\gamma}_u$ and γ_u over the length of the data set were calculated, and the corrected L_{psv} is solved for:

$$L_{psv} = -3 + 10log_{10}\left(\frac{1}{P}\sum_{j=1}^{P}\frac{E[\gamma_{uj}]}{E[\bar{\gamma}_{uj}]}\right) \tag{29}$$

where the -3 in this equation is the assumed difference in transmission power in dB [18] and P is the number of data sets used in calculating L_{psv} for each satellite. The average was $P = 12.2$, with the fewest number of data sets being 9 for PRN 29. The resulting L_{psv} values are presented in Table I for every satellite in the GPS constellation as of February, 2013. Also presented are their standard deviations.

The smallest power decrement was -2.32 dB for PRN 3, and the largest power decrement is -2.93 dB for PRN 4. These results also illustrate that the algorithm is operating correctly in the absence of spoofing, as the un-normalized spoofing detection statistic closely matches the mean value predicted from the C/A code carrier-to-noise ratio after applying this small correction. Had the corrected L_{psv} values been wildly different from -3 dB, one might have questioned the validity of the analysis on which this entire spoofing detection technique is based.

The values presented in Table I are subject to change for a variety of reasons. These include the addition or removal of satellites into the active GPS constellation, varying P(Y) code transmission power ("flex power" [18]), or switching between different transmitters on the satellite. These values should be monitored continuously and updated as needed.

Detection of a Spoofing Attack

To test the capability of this system to detect spoofing, a sophisticated GPS spoofer [2, 3, 26, 27] has been used to conduct a number of attacks in a controlled environment. For each test, the output of the GPS spoofer was combined electrically with the signal coming from the roof-mounted antenna. This was done to ensure that no harmful signals were transmitted over the air.

Consider a representative test that was conducted on July 14, 2011, with the spoofer co-located with the defended receiver in Austin. Fig. 3 illustrates this attack, with a signal that is being spoofed shown in the top panel, and an un-spoofed signal shown in the bottom panel. For both figures, the normalized spoofing detection statistic, γ_{norm} is the solid blue line, its expected value $\bar{\gamma}_{norm}$ is the dash-dotted green line, and the spoofing detection threshold, γ_{th} is the dashed red line.

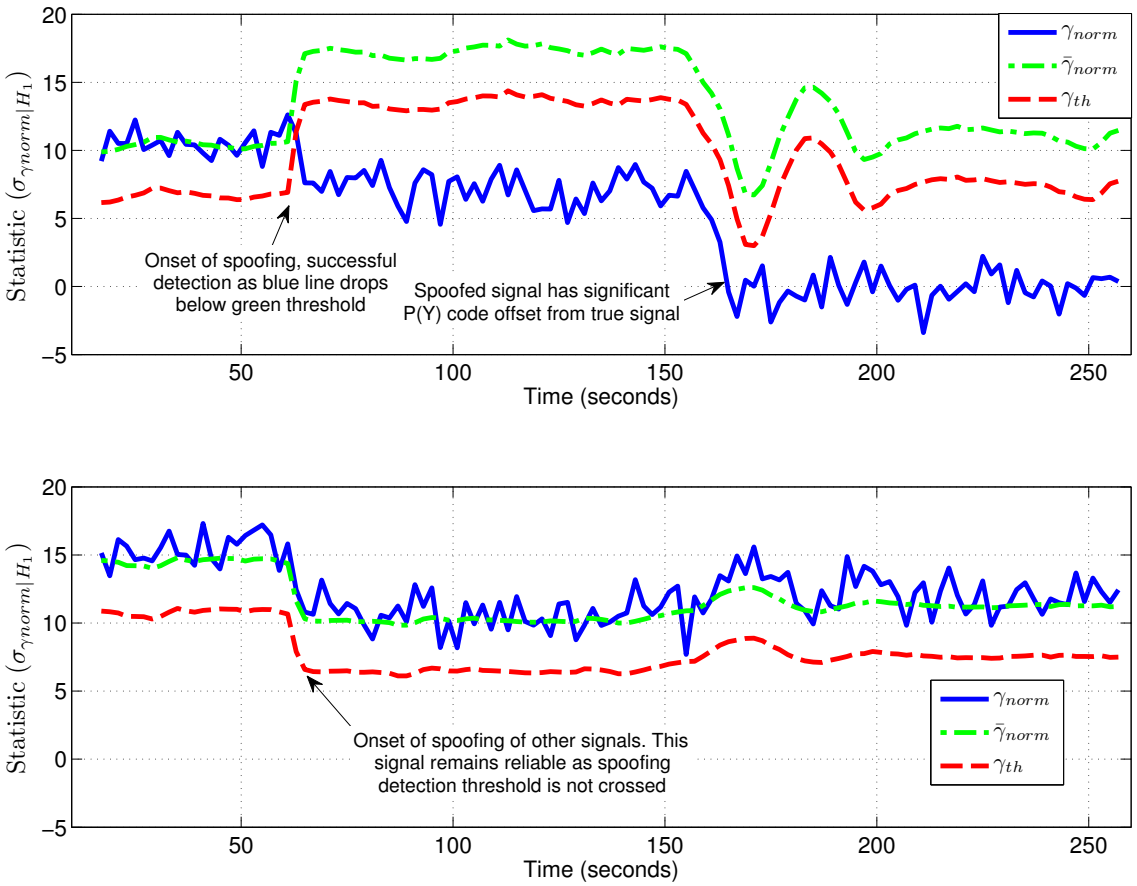

Fig. 3: Normalized spoofing detection statistic and related quantities for a spoofed signal (top panel) and an un-spoofed signal (bottom panel) during a spoofing attack.

During this test, there was no spoofing of any signal for the first 60 seconds, to establish a baseline for normal operation. At 60 seconds, spoofing of some of the signals began, but the spoofer transmitted its best estimate of the "true" signal, though it did not maintain phase coherence with the true signal. For the spoofer to capture the target receiver, the transmitted power must be larger than the power received from the satellite. This increase in power has the effect of reducing the gain in the receiver due to the action of its automatic gain control. For the spoofed signal, this reduction in gain causes the P(Y) code cross-correlation to fall. The received C/A code power, on the other hand, rises due to the increased power of the spoofer. This rise causes $\bar{\gamma}_{norm}$ and γ_{th} to rise. These two effects combine to cause γ_{norm} to cross below the spoofing detection threshold, triggering an alarm. Thus, even though the spoofer was not "lying"to the receiver about its location, it was still detected by virtue of its higher C/A code power level and the lower power level of the P(Y) code.

For the un-spoofed signal shown in the bottom panel, the reduced gain acted equally on both the C/A and P(Y) code signal. Thus the cross-correlation detection statistic's expected value remained accurate and no spoofing was

detected.

Approximately 90 seconds later, at receiver time 150 seconds, the spoofer began moving the spoofed signal away from the true signal, and the spoofing detection statistic for the spoofed signal became zero mean, as would be expected once the spoofed C/A code drags the defended receiver timing far away from the true P(Y) code.

The actual spoofing detection output for all signals, $\tilde{\gamma}$ from Eq. 27, during this same spoofing attack is shown in Fig. 4. It is quite clear that PRNs 23 and 30 remained free of spoofing, while all the other signals were spoofed from 60 seconds onward. These results are consistent with the spoofing test parameters.

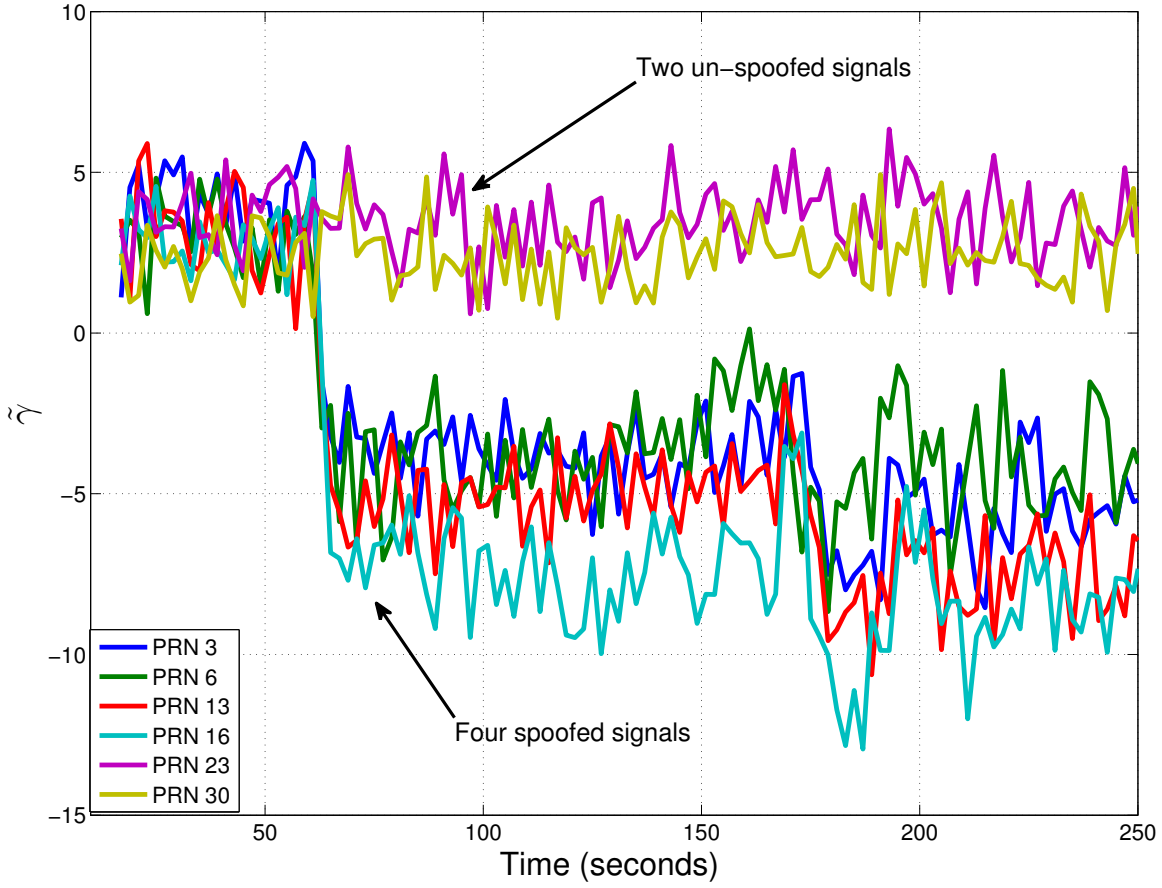

Fig. 4: Receiver output for all signals during a spoofing attack.

The probability of successful spoofing detection for all signals during this spoofing attack is shown in Fig. 5. This probability is quite high for most of the signals for the duration of the test. PRN 30 had a significantly lower detection probability after the onset of the spoofing attack at 60 seconds. After this time, the carrier-to-noise ratio of the C/A code for this PRN was quite low, approximately 38 dB-Hz, which is the cause of this low detection probability. This "problem" occurs because PRN 30 is not being spoofed, which gives it a lower carrier-to-noise ratio. Had it been spoofed with a signal strong enough to drag off the receiver tracking loops, the problem would

not have occurred, and the actual spoofing attack would likely have been detected. The drop in P_d seen on several signals at approximately 160 seconds is due to a rapid drop in the carrier-to-noise ratio of those signals, presumably due to the spoofed and true C/A codes briefly interfering with each other as the spoofer drags the receiver away from the true signal.

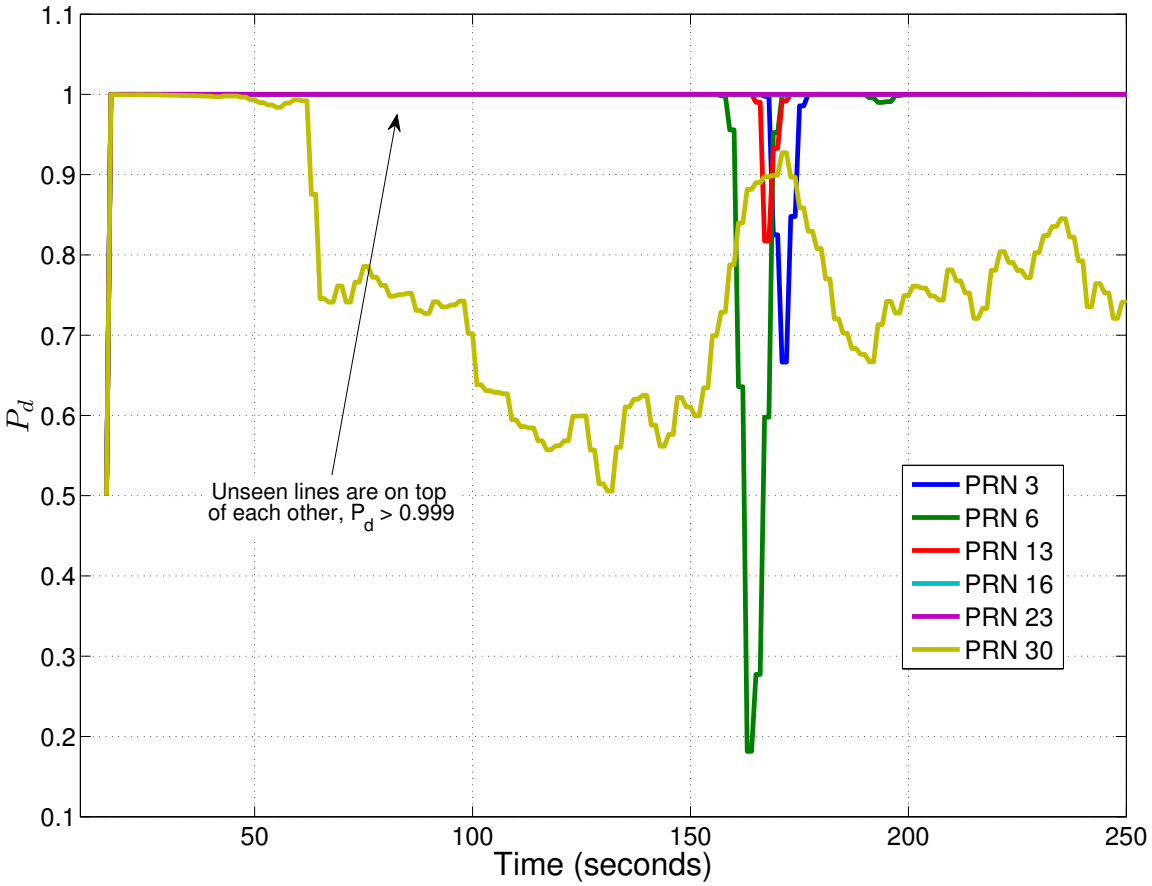

Fig. 5: Probability of detection for all signals during a spoofing attack.

These results and many similar tests represent the first detections of spoofing attacks in a real-time system using a single antenna per receiver. This also represent the first real-time implementation of the cryptographic spoofing detection technique of Refs. [12, 13, 14, 15].

Summary and Conclusion

A method for detecting spoofing of the GPS L1 C/A code signal has been implemented in a real-time system. This method assumes that the encrypted P(Y) code signal is free of spoofing. This assumption allows the use of a "reference receiver" that is free of spoofing to assist in detection of possible spoofing at a "defended receiver". If only the C/A code is being spoofed, it is possible to do a cross-correlation with the portion of the data from

the two receivers that is in quadrature with the C/A code. The code and carrier phase relationship between the C/A and P(Y) code signals are known, and the portion of the signal containing the P(Y) code can be isolated and used for cross-correlation. If there is no spoofing at the defended receiver, this cross-correlation will result in a large spoofing detection statistic due to the P(Y) code autocorrelation properties. If the defended receiver is being spoofed, this has the effect of introducing a code phase offset between the spoofed C/A code and the un-spoofed P(Y) code, resulting in a small spoofing detection statistic.

A hypothesis test is constructed to allow determination of a threshold for the spoofing detection statistic. This threshold guarantees a low probability of false alarm. For sufficiently strong signals and sufficiently long detection intervals, the probability of detection is very close to one.

The new method has been tested by subjecting it to realistic spoofing attacks. Successful detection of these spoofing attacks has been demonstrated. Nominal receiver response in the absence of spoofing has also been demonstrated in a number of tests.

As a side benefit of this work, the power differences between the C/A and P(Y) code signals has been investigated experimentally. The actual differences in power levels between these two signals can vary by more than a decibel from the value implied in the system documentation. These power difference calibrations have been used to develop the precise hypothesis test analysis on which the new spoofing detection method is based.

REFERENCES

[1] J. S. Warner and R. G. Johnston, "A simple demonstration that the Global Positioning System (GPS) is vulnerable to spoofing," *Journal of Security Administration*, 2003.

[2] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner, Jr., "Assessing the spoofing threat: development of a portable GPS civilian spoofer," in *Proceedings of the ION GNSS Meeting*, (Savannah, GA), Institute of Navigation, 2008.

[3] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner, Jr., "Assessing the spoofing threat," *GPS World*, vol. 20, pp. 28–38, Jan. 2009.

[4] D. Shepard, J. Bhatti, and T. Humphreys, "Drone hack: Spoofing attack demonstration on a civilian unmanned aerial vehicle," *GPS World*, vol. 23, pp. 30–33, Aug. 2012.

[5] Anon., "Vulnerability assessment of the transportation infrastructure relying on the Global Positioning System," tech. rep., John A. Volpe National Transportation Systems Center, 2001.

[6] L. Scott, "Anti-spoofing and authenticated signal architectures for civil navigation systems," in *Proceedings of the ION GNSS Meeting*, (Portland, Oregon), pp. 1542–1552, Institute of Navigation, 2003.

[7] S. Peterson, "Iran hijacked US drone, says Iranian engineer," 2011. http://www.csmonitor.com/World/Middle-East/2011/1215/Exclusive-Iran-hijacked-US-drone-says-Iranian-engineer-Video.

[8] J. S. Warner and R. G. Johnston, "GPS spoofing countermeasures," Dec. 2003. http://www.homelandsecurity.org/bulletin/DualBenefit/warner_gps_spoofing.html.

[9] N. White, P. Maybeck, and S. DeVilbiss, "Detection of interference/jamming and spoofing in a DGPS-aided inertial system," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 34, no. 4, pp. 1208–1217, 1998.

[10] P. Y. Montgomergy, T. E. Humphreys, and B. M. Ledvina, "Receiver-autonomous spoofing detection: Experimental results of a multi-antenna receiver defense against a portable civil GPS spoofer," in *Proceedings of the ION ITM*, (Anaheim, CA), Jan. 2009.

[11] M. L. Psiaki, S. P. Powell, and B. W. O'Hanlon, "GNSS spoofing detection by correlating carrier phase with rapid antenna motion," *GPS World*, vol. 24, pp. 53–58, June 2013.

[12] P. Levin, D. De Lorenzo, P. Enge, and S. Lo, "Authenticating a signal based on an unknown component thereof," June 2011. US Patent 7,969,354 B2.

[13] B. O'Hanlon, J. Bhatti, T. E. Humphreys, and M. Psiaki, "Real-time spoofing detection using correlation between two civil GPS receiver," in *Proceedings of the ION GNSS Meeting*, (Nashville, Tennessee), pp. 3584–3590, Institute of Navigation, 2012.

[14] M. L. Psiaki, B. W. O'Hanlon, J. A. Bhatti, and T. E. Humphreys, "Civilian GPS spoofing detection based on dual-receiver correlation of military signals," in *Proceedings of the ION GNSS Meeting*, (Portland, Oregon), pp. 2619–2645, Institute of Navigation, 2011.

[15] M. Psiaki, B. W. O'Hanlon, J. Bhatti, and T. E. Humphreys, "GPS spoofing detection via dual-receiver correlation of military signals," *IEEE Transactions on Aerospace and Electronic Systems*, 2012. In press.

[16] K. Wesson, M. Rothlisberger, and T. E. Humphreys, "Practical cryptographic civil GPS signal authentication," *NAVIGATION, Journal of the Institute of Navigation*, vol. 59, no. 3, pp. 177–193, 2012.

[17] T. E. Humphreys, "Detection strategy for cryptographic GNSS anti-spoofing," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 49, pp. 1073–1090, April 2013.

[18] Anon., "IS-GPS-200G: Navstar GPS space segment/navigation user interfaces," tech. rep., Science Applications International Corporation, 2012. http://www.gps.gov/technical/icwg/IS-GPS-200G.pdf.

[19] C. Edgar, J. Price, and D. Iteigh, "GPS block IIA and IIR received signal power measurements," in *Proceedings of the ION NTM*, (Long Beach, CA), pp. 401–411, Institute of Navigation, Jan. 1998.

[20] C. Edgar, D. B. Goldstein, and P. Bently, "Current constellation GPS satellite ground received signal power measurements," in *Proceedings of the ION NTM*, (San Diego, CA), pp. 948–954, Institute of Navigation, Jan. 2002.

[21] M. L. Psiaki and B. W. O'Hanlon, "System identification of a GNSS receiver's RF filter impulse response function," in *Proceedings of the ION GNSS Meeting*, (Portland, Oregon), pp. 3690–3708, Institute of Navigation, 2011.

[22] Anon., "Recommendation for key management—Part I: General (revision 3)," sp 800-57, National Institute of Standards and Technology, July 2007.

[23] B. O'Hanlon, M. Psiaki, S. Powell, J. Bhatti, T. E. Humphreys, G. Crowley, and G. Bust, "CASES: A smart, compact GPS software receiver for space weather monitoring," in *Proceedings of the ION GNSS Meeting*, (Portland, Oregon), Institute of Navigation, 2011.

[24] B. M. Ledvina, "Efficient real-time generation of bit-wise parallel representations of oversampled carrier replicas," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, Oct. 2011.

[25] B. M. Ledvina, M. L. Psiaki, S. P. Powell, and P. M. Kintner, Jr., "Bit-wise parallel algorithms for efficient software correlation applied to a GPS software receiver," *IEEE Transactions on Wireless Communications*, vol. 3, Sept. 2004.

[26] T. E. Humphreys, J. A. Bhatti, D. P. Shepard, and K. D. Wesson, "The Texas spoofing test battery: Toward a standard for evaluating GNSS signal authentication techniques," in *Proceedings of the ION GNSS Meeting*, (Nashville, Tennessee), Institute of Navigation, 2012.

[27] D. P. Shepard, T. E. Humphreys, and A. A. Fansler, "Evaluation of the vulnerability of phasor measurement units to GPS spoofing attacks," *International Journal of Critical Infrastructure Protection*, vol. 5, no. 3-4, pp. 146–153, 2012.

PRN	Mean (dB)	σ (dB)	PRN	Mean (dB)	σ (dB)
1	-2.87	0.04	17	-2.92	0.03
2	-2.92	0.04	18	-2.88	0.05
3	-2.32	0.13	19	-2.83	0.07
4	-2.93	0.05	20	-2.82	0.06
5	-2.92	0.03	21	-2.91	0.08
6	-2.86	0.05	22	-2.91	0.03
7	-2.90	0.06	23	-2.89	0.05
8	-2.90	0.04	24	-2.80	0.08
9	-2.83	0.05	25	-2.88	0.06
10	-2.87	0.06	26	-2.78	0.14
11	-2.89	0.08	27	-2.84	0.05
12	-2.87	0.04	28	-2.61	0.02
13	-2.88	0.04	29	-2.89	0.03
14	-2.89	0.03	30	-2.88	0.05
15	-2.89	0.03	31	-2.89	0.03
16	-2.92	0.03	32	-2.83	0.04

TABLE I: Observed decrement in transmitted power between L1 C/A and L1 P(Y) signals, by satellite.